IBM ILOG OPL V6.3

IBM ILOG OPL Release Notes

© Copyright International Business Machines Corporation 1987, 2009
US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Copyright

COPYRIGHT NOTICE

© Copyright International Business Machines Corporation 1987, 2009.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Trademarks

IBM, the IBM logo, ibm.com, WebSphere, ILOG, the ILOG design, and CPLEX are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at *http://www.ibm.com/legal/copytrade.shtml*

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Acknowledgement

The language manuals are based on, and include substantial material from, The OPL Optimization Programming Language by Pascal Van Hentenryck, © 1999 Massachusetts Institute of Technology.

Table of contents

Release Notes.....5
Meet IBM ILOG OPL.....7
New and changed in OPL V6.3.....8
New and changed in OPL 6.2.....9
New and changed in OPL 6.1.1.....13
New and changed in OPL 6.1.....14
New and changed in OPL 6.0.1.....17
New and changed in OPL 6.0.....19
The OPL IDE.....21
Solving scheduling problems with constraint programming.....23
Language.....24
.NET interfaces.....27
Databases.....29
oplrun.....30
Examples.....31
Port changes.....32
New and changed in OPL 5.5.....35
Deprecated.....37
CPLEX.....38
IDE.....39
Language.....40

© Copyright IBM Corp. 1987, 2009

Scripting....41
oplrun....42
Interfaces....43
Examples and model library....44
New and changed in OPL 5.2....47
Support for constraint programming....49
Deprecated....51
Licensing - trial version....53
Performance....54
Localization....55
Terminology....56
Changes to the language....57
Changes to the IDE....58
Examples and models....60
Changes to the documentation set....62
New and changed in OPL 5.1....63
Deprecated features....64
Performance and new ports....66
Changes to the language....67
Changes to the IDE....68
oplrun....70
Miscellaneous....71
Incompatibilities....72
New and changed in OPL 5.0....73
Deprecated features....74
Support for ILOG ODM....75
Changes to the language....77
Changes to the modeling language....78
ILOG Script....80
oplrun....81
Changes to the IDE....83
The Main window....84
Project management and execution....87
Changes to the documentation....88
Index....89

Release Notes

Contains an overview of the main changes in this release and previous releases. If you are upgrading from a version of OPL earlier than OPL 6.0, you also need to read the section *Migrating from previous versions of OPL* in the *OPL Migration Guide*.

In this section

Meet IBM ILOG OPL
Briefly introduces the components of IBM ILOG OPL.

New and changed in OPL V6.3
This release has mainly concentrated on polishing existing OPL features. It does contain some new language features.

New and changed in OPL 6.2
These release notes highlight improvements and new or obsolete features in OPL 6.2.

New and changed in OPL 6.1.1
OPL 6.1.1 is a technical release that corrects some known errors.

New and changed in OPL 6.1
These release notes highlight improvements and new or deprecated features in OPL 6.1.

New and changed in OPL 6.0.1
Lists the problems fixed by OPL 6.0.1, which is a technical update to OPL 6.0.

New and changed in OPL 6.0
These release notes highlight improvements and new features in OPL 6.0. Please read these notes before using OPL 6.0.

© Copyright IBM Corp. 1987, 2009

New and changed in OPL 5.5

These release notes highlight improvements and new features in OPL 5.5. Please review these notes before using OPL 5.5.

New and changed in OPL 5.2

These release notes highlight improvements and new features in OPL 5.2. Please review these notes before using OPL 5.2. The main change in OPL 5.2 over OPL 5.1 is the introduction of constraint programming features.

New and changed in OPL 5.1

OPL 5.1 is a minor update to OPL 5.0. In this release, you will benefit from more ports, improved performance, external function calls, table loading, `dexpr` arrays, scripting in data files, better usability in the profiler, bug report packaging, localization, and more.

New and changed in OPL 5.0

OPL 5.0 is a major update to OPL 4.x. It is also built upon ILOG Concert Technology. In addition to significant gains of performance, OPL 5.0 brings the changes listed in these release notes.

Meet IBM ILOG OPL

Thank you for your interest in IBM® ILOG® OPL, a powerful system for rapid development and deployment of optimization applications.

IBM ILOG OPL (or OPL for short) consists of:

- the Optimization Programming Language (OPL) for developing optimization models
- an Integrated Development Environment (IDE) to execute and test optimization models
- a command line tool (`oplrun`) to execute models from the command line
- Application Programming Interfaces (APIs) to embed models into standalone applications
- support for IBM ILOG Optimization Decision Manager (ODM)

Before you start working with OPL, we suggest you read:

- The `readme` file, which contains additional information and should be consulted before launching the IDE.
- *New and changed in OPL V6.3* in these *Release Notes*, for an overview of how this release improves on previous releases.
- *Migrating from previous versions of OPL* in the *OPL Migration Guide*, if you are upgrading from an earlier release than OPL 6.0.
- How to read the OPL documentation for details of prerequisites, conventions, documentation formats, and other general information.

Note: The Release Notes for the versions of OPL other than the current one remain accessible from the table of contents. However, the links they provide to the documentation are unclickable textual links, not hypertext links, because the structure of the documentation set has changed in the meantime. Those links refer to the documentation set actually delivered with the corresponding version of the product, not to the documentation set of the current release.

New and changed in OPL V6.3

- *Language*
- *Database Management*
- *Performance improvement*

Language

New features

The following OPL functions are available to obtain values from constraints in a post-processing context:

- dual
- slack
- reducedCost

They are documented in the Language Quick Reference.

Language changes

- The names `dual` and `slack` are no longer accepted as identifiers as they are now function names.
- OPL now raises errors for undefined external data (syntax `=...`) even when that data is not used by the model.

Database Management

OPL V6.3 uses DB Link V5.3 for the processing of Relational Database Management Systems (RDBMS).

Performance improvement

On AIX systems, OPL 6.3 is 15% faster than the previous release.

New and changed in OPL 6.2

This release has concentrated on polishing existing features, and improving stability and usability. The debugging feature is more robust and freeze situations during debugging have been fixed.

Some new features appear in the OPL IDE and the overall quality of the IDE has improved significantly.

- *OPL IDE*
- *Language*
- *OPL runtime*
- *Examples*
- *Obsolete in OPL 6.2*

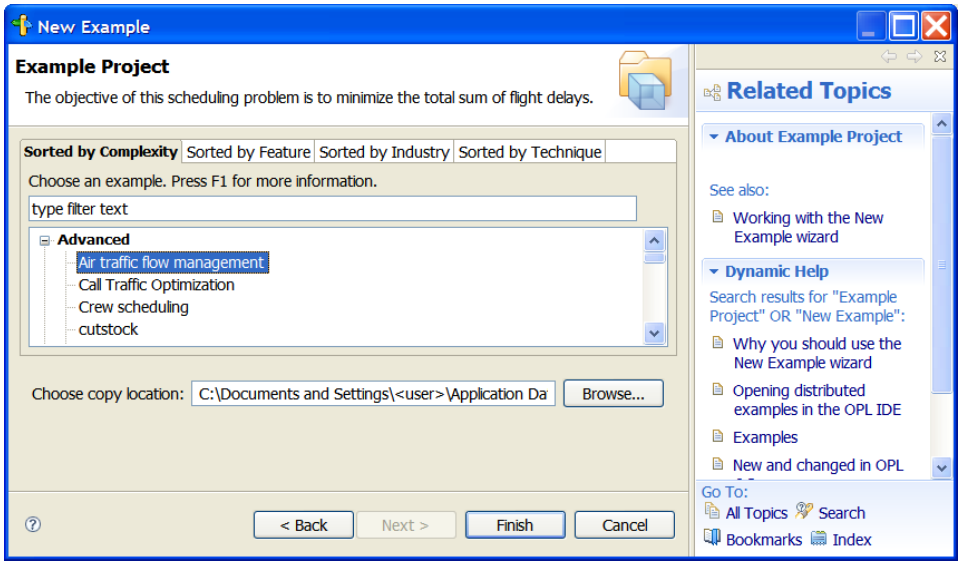
OPL IDE

The OPL IDE has new icons and an updated model outline as well as the New Example wizard, and Model Completion and Model Templates.

New method of importing distributed examples

When opening the examples distributed with OPL 6.2, users now have the option of opening and working with them in the OPL IDE using the New Example wizard. This offers many benefits to the user:

- Always working with a copy, not the original — Opening examples using the New Example wizard ensures that you are always working with a copy of the distributed example. Therefore, if you make changes and save them, the original example is always available to you in its original form.
- Flexibility of project destination — Using the New Example wizard, you can specify any folder in your file system as the destination for the project.
- Enhanced project descriptions — Using the New Example wizard, you see longer, more explicit descriptions of each project.
- Enhanced sorting and display options — The New Example wizard contains multiple tabs that allow you to view the examples sorted by complexity of the example, feature being demonstrated in the example, industry represented, and mathematical programming technique being used.
- Built-in Help — By selecting an example and pressing **F1** or clicking the Help icon you can display the documentation for that example in the New Example wizard itself.



For more information, see *Opening distributed examples in the OPL IDE* in *Getting Started with the OPL IDE*.

New Model Completion and Model Templates features provide code assistance in the OPL IDE editor

Model Completion is a feature of the OPL IDE editor that allows it to prompt you during the creation of an OPL model, and allows you to select OPL and ILOG Script keywords and functions from a context-sensitive list, and insert them into your model. You access this level of code assistance by pressing **Ctrl + space** in the OPL IDE editor.

Another level of code assistance is available when 1) your model contains tuples, and 2) an Outline is present for the model. This level of code assistance prompts you with variables and tuple component names from the model Outline; and is accessed using a dot syntax. That is, you access it by typing a single period or dot (**.**) in the OPL editor.

Model Templates is a related feature of the OPL IDE editor that allows you to select longer snippets of code, along with placeholders for required variables, from a context-sensitive list, and insert them into your model. As with Model Completion, you access this by pressing **Ctrl + space** in the OPL IDE editor.

For more information, see *Model Completion and Model Templates* in the *IDE Reference Manual*.

New icons

To enhance the usability and visibility of the different functions and features a complete new set of icons has been introduced.

Improved engine support

- Parallel CPLEX is available for solving optimization problems from within the IDE.
- In the CP Optimizer

- The `alternative` constraint has been generalized to synchronize more than one alternative interval variable with a master interval variable, a feature than can be useful for multi-skill problems for instance.
- Cumulative functions can be constrained by an integral expression or variable, allowing resource capacity (or minimum level) to be an unknown of the problem.

Changed behavior in the IDE with respect to the solution pool
In OPL 6.2, you may observe fewer solutions in the pool than in previous versions of OPL for the same model with the same data. That is because OPL calls CPLEX `populate` only instead of `solve` + `populate`.

Depending on the model, the number of solutions generated for the new OPL setting `populate` can be equal to, greater than, or smaller than, the number of solutions generated for the old OPL setting `solve` + `populate`.

To obtain the previous behavior, set the CPLEX parameter `solution pool intensity` to "very aggressive", which is `cplex.solnpoolintensity=4`.

Language

- Messages sent by the type checker have been improved and include user defined names.
- There is a restriction on using `dexpr` with patterns.

  If you declare a `dexpr` with patterns and keys:

  - if you declare it with all the pattern components, you must use it with all the components
  - if you declare it with only the keys, you must use it with the keys only

  You cannot combine a declaration of all the patterns and a usage of only the keys.

OPL runtime

The IBM JRE is supported in this release.

Note that the solution is invalidated if the engine is used for another solve. Post-processing elements in `thisOplModel` may become unavailable.

Examples

- The OPL cutting stock example, `cutstock`, has been improved to make better use of `IloOplDataElements` to demonstrate sharing data between models.
- New examples have been added to the standard distribution in `<OPL_install>\examples\opl\models`
  - `TravelingSalesmanProblem`
  - `CallTrafficOptimization`
  - `Portfolio`, to demonstrate portfolio optimization with bender's decomposition

Obsolete in OPL 6.2

The `endall()` methods are no longer supported in OPL 6.2.

- **C++**

  The methods `IloOplModel::endAll()` and `IloOplRunConfiguration::endAll()` were deprecated in a previous version of OPL and have been removed from OPL 6.2. Use `IloOplModel::end()` or `IloOplRunConfiguration::end()` instead.

- **Java, ILOG Script**

  The methods `IloOplModel.endAll()` and `IloOplRunConfiguration.endAll()` were deprecated in a previous version of OPL and have been removed from OPL 6.2. Use `IloOplModel.end()` or `IloOplRunConfiguration.end()` instead.

- **.NET**

  The methods `OplModel.EndAll()` and `OplRunConfiguration.EndAll()` were deprecated in a previous version of OPL and have been removed from OPL 6.2. Use `OplModel.End()` or `OplRunConfiguration.End()` instead.

New and changed in OPL 6.1.1

OPL 6.1.1 is a technical release that corrects some known errors.

The known issues that have been corrected can be found on the *ILOG Customer Support site*. You will need your maintenance reference number to access the site.

New and changed in OPL 6.1

This release has concentrated on polishing existing features, and many usability issues and bugs have been fixed.

Windows 64 bits

- OPL IDE

  The OPL IDE is supported on Microsoft Windows 64 bits.

  Note that on 64-bit Windows there is no Microsoft-provided ODBC driver, so Microsoft Access is not available. Microsoft Excel is supported, but not through ODBC.

- .NET API

  The OPL interfaces for .NET are now supported on Microsoft Windows 64 bits.

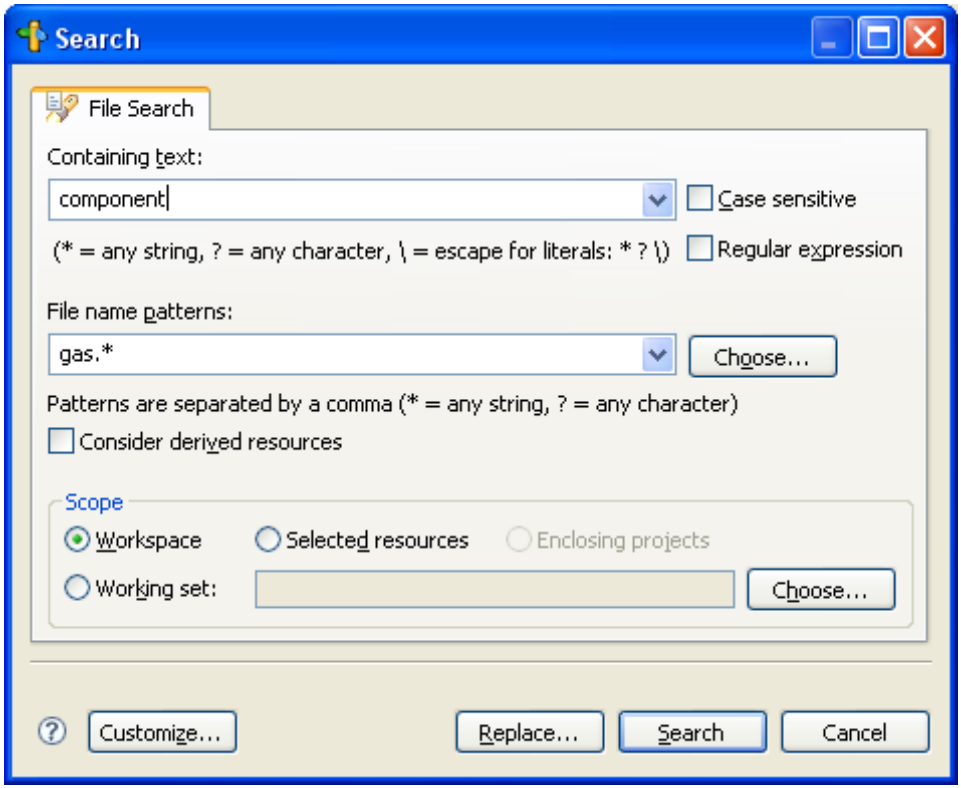
  For details, please consult the file `mptable.html`, distributed with OPL. This file can be found in the root installation directory: `<OPL_dir>\mptable.html`.

OPL IDE

Several improvements have been made to the OPL IDE; the main ones are described here.

- Searching in a file

  The Editing Area incorporates more powerful search capabilities than were present in previous releases of OPL. You can search a single file or multiple files at once, and by a large number of criteria. The menu option, or the toolbar button, **Search** displays the File Search dialog box.



For more information, see *Searching files* in the *IDE Reference Manual*.

- Data views

  There are more possible data views, in particular for the following scheduling objects:

  - `cumulFunction` arrays
  - `stateFunction` arrays
  - sequences
  - sequence arrays

  Data views can be sorted by column, even for arrays.

- Memory allocation

  You can change the amount of memory allocated to the OPL IDE in the file `<OPL_dir>\oplide\oplide.ini`.

  For more information, see *OPL IDE memory allocation* in the *IDE Reference Manual*.

Language

- Function names can no longer be used as tuple component names. For example, the function name `before` is now refused for tuple components.
- The undocumented `ind()` function no longer exists. The boolean-to-int cast is now automatic in OPL.
- There is a new scripting method `IloTupleSet.addOnly()`, which is faster than `IloTupleSet.add()`.

Interfaces

The same data source can now be used for multiple models.

oplrun

- `oplrun -ph` <project-name>

  This new option (project help) displays the run configurations available in the specified project. Example: `oplrun -ph mynew.prj` displays all the configurations in `mynew.prj`.

- `oplrun -h`

  The output of this command has been reorganized.

Deprecated in OPL 6.1

JDK 5.0 is deprecated in OPL version 6.1 and will not be supported in the next version of OPL.

New and changed in OPL 6.0.1

Microsoft Windows support

OPL 6.0.1 supports Windows Server 2003, as well as Windows XP and Windows Vista.

Known issues corrected in technical release 6.0.1

- Improved error message to indicate that you cannot slice on sets with nested constraints on subtuples.
- The CPLEX extraction for `in` in a `forall` statement now gives the correct result.
- Arrays contained in a tuple are now correctly initialized.
- The IDE editor is now faster when editing a large file.
- Boolean values are now refused for the `size` property of `dvar` intervals.
- The extraction is now correct when a model is modified via a script.
- There is now a correct evaluation of dual and slack when the constraint is not a range with an upper bound and a lower bound, but has only a lower bound or an upper bound.
- There is no longer a problem when a complex `dexpr` in a submodel is evaluated.
- In scripting, assigning an `expr` to an `int` or a `num` now raises an error.
- Expressions `heightAtStart` and `heightAtEnd` now work when used in aggregations.
- If, after generating an ODM application, an error is introduced and fixed, the error no longer persists.
- The IDE no longer crashes when using an invalid PWL statement in a `forall` loop followed by a named constraint.
- The IDE no longer crashes when slicing on a union of two tuple sets.
- There is no longer a `NullPointerException` when you click on the decision variables in the Problem Browser in a QP model.
- You can now indicate, in the IDE preferences, a maximum size for a loaded data file.
- CP parameters set through a script no longer cause the IDE to crash.
- There are no longer internal errors and inconsistent results after solving a scheduling model.

New and changed in OPL 6.0

These release notes highlight improvements and new features in OPL 6.0. Please read these notes before using OPL 6.0.

In this section

The OPL IDE
Summarizes new and changed features in the OPL IDE.

Solving scheduling problems with constraint programming
Describes the OPL support for constraint programming used in both combinatorial and scheduling problems.

Language
Describes changes to the OPL language.

.NET interfaces
Describes changes to the .NET interfaces for OPL.

Databases
Describes changes to database support in this release.

oplrun
Describes changes to the `oplrun` command.

Examples
Describes changes to the OPL examples.

Port changes
Lists new and deprecated ports for OPL 6.0.

The OPL IDE

In OPL Development Studio 6.0, the IDE has been thoroughly redesigned to increase productivity and usability for modeling, solving and debugging of optimization models.

Improvements and changes have been made to:

- project management
- model editing
- debugging

and many other key aspects of interactivity with OPL.

The changes are described below in more detail. Due to the redesign, the projects are stored differently and it is therefore necessary to migrate OPL 5.x projects to OPL 6.0. Instructions for migration are included below.

New and changed features in the OPL IDE

The new and changed features are described in detail in the documentation and are summarized briefly in these Release Notes.

- *Getting Started with the OPL IDE* explains how to work with files in the IDE and how to create and execute an OPL project.
- *The IDE Tutorials* illustrate the features of the IDE through examples.
- The *IDE Reference Manual* describes all the elements of the OPL graphical user interface.

Redesigned IDE for faster model development

- New text editor with powerful navigation, search and replace
- File history, allowing comparison of versions during model development
- Flexible side-by-side file comparison, across projects
- Dynamic help facility: the help content dynamically changes as the user types

Improved way of managing recent solves and run configurations

- Remembering the most recent launches for run configurations across projects
- Fast launch of a run configuration through the contextual menu

Team collaboration

- Integration with source code control systems (CVS is pre-packaged)
- Enables teams to work jointly on shared models and other project artifacts

Performance gains

- Interaction responsiveness is faster (for example, huge named constraints, or big arrays, are no longer slow in getting back the focus of the IDE).
- Performance improvements (especially on Linux platforms)

Other changes and migration notes

- `.prj` files are deprecated and are replaced by `.oplproject` files. An OPL project now corresponds to the folder containing that file. Refer to the *OPL 6.0 Migration Guide* for details.
- Conflicts and Relaxations are now displayed in separate output tabs
- The `oplrun` command is now available from the Run menu:

  Run>External Tools>Run As>oplrun
- Profiler data is not available after an aborted solve, but will be added again in the next release
- Feasible solutions are displayed in the Solutions tab only if the options **Postprocess feasible solutions** and **Display solution** are selected in a settings file, under **Language>Run**.
- OPL 5.x projects must be migrated to OPL 6.0. The migration can be done via the File menu in the OPL IDE: **File>Import>Migrate OPL 5.x projects**. The Import wizard provides several options for storing the migrated versions. The *OPL 6.0 Migration Guide*, distributed with OPL 6.0 documentation, will take you through the detailed steps of migrating your existing 5.x projects.
- The example projects provided with the OPL 6.0 distribution are in the directory:

  `<OPL_dir>\examples\opl`

  These examples are accessed via the command:

  File>Import>Existing OPL 6.x projects.

  To import, for example, the warehouse project, you would select the root directory `warehouse` from `<OPL_dir>\examples\opl`.

Solving scheduling problems with constraint programming

ILOG OPL Development Studio 6.0 supports constraint programming for both combinatorial and scheduling problems, completing the reintroduction of constraint programming that was started in OPL 5.2, and providing the same modeling capabilities for CP models as there is in OPL 3.7.

Scheduling support

ILOG CP Optimizer 2.0 has been extended with algorithms specialized in solving and optimizing the constraints and costs that are specific to scheduling problems, such as, to name just a few:

- cumulative constraints that limit the use of a resource
- precedence constraints between tasks
- resource state management and associated setup times or earliness and tardiness costs

The language of CP Optimizer for ILOG OPL Development Studio has been enriched with a new variable type, `intervals`, and constraints that apply to these intervals.

The associated documentation is:

- *Scheduling with IBM ILOG OPL*
- The *Scheduling* section in the *Language Reference Manual*

The examples can be found in the directories:

```
<OPL_dir>\examples\opl\sched_*
```

```
<OPL_dir>\examples\opl\models\sched_*
```

Comparison with OPL 3.7

The support of constraint programming in ILOG OPL Development Studio relies on powerful default search algorithms that enable you to use the 'model and run' development process. If necessary, you can control these search algorithms with parameters or declarative search phases.

The basic entities of the language for scheduling support are interval variables and functions. They involve a smaller number of concepts and provide more flexibility to express the constraints of scheduling models. For a guide to translating OPL 3.7 scheduling models, refer to the document:

OPL 6.0 Migration Guide, Migration from OPL 3.x (CP Scheduling projects)

Language

New keywords

These new keywords are no longer accepted as identifiers.

- `pwlFunction, pwlfunction`
- `stepFunction, stepfunction`
- `stepwise`

These new CP keywords are still accepted as CPLEX identifiers.

- `interval`
- `sequence`
- `size`
- `intensity`
- `optional`
- `stateFunction, statefunction`
- `cumulFunction, cumulfunction`

Identifiers

Certain names are no longer allowed to be used as identifiers

- Function names are no longer allowed to be identifiers.
- The algorithm names CP, CPLEX, `IloCplex`, `IloCP` are now keywords and are therefore refused as element identifiers.

Multi-criteria search phases

In OPL 5.2 and 5.5, search phases were limited. The engine would branch on variables and then on values. Now, in OPL 6.0, this limit no longer exists and the two phases are merged into one.

For example, the search phase we have in the `carseq` project:

```
var phase1 = f.searchPhase(slot,
                           f.selectSmallest(f.varIndex(slot)),
                          f.selectLargest(f.explicitValueEval(values, valueEval,
 0)));
```

could be changed into

```
var multiPhaseVar = new Array( f.selectSmallest( f.domainSize() ), f.
selectRandomVar() );
               var multiPhaseValue = new Array( f.selectSmallest( f.value() )
, f.selectRandomValue() );
var phase1 = f.searchPhase( multiPhaseVar, multiPhaseValue );
```

Note that this is just an example and would not make the `carseq` example more efficient.

Sets that are iterated on cannot be modified

If you write in a script `for(a in s)`, you cannot modify `s` .

The following code is not accepted any more:

```
tuple t
{
  int x;
  int y;
}

{t} s={<1,2>,<2,2>};

 execute
{
  for(var i in s) i.y=i.x;
}
```

and should be rewritten as:

```
tuple t
{
  int x;
  int y;
}

{t} s={<1,2>,<2,2>};

execute
{
  for(var k=0;i<=s.size;k++)
  {
    var i=Opl.item(s,k);
    i.y=i.x;
  }
}
```

Tuple sets with multiple indirections (with keyword)

Slicing is not allowed on `tupleSet` with an indirection of `subTuple` on another set with indirection.

The following code contains an example, in the last line, of what you are no longer allowed to write.

```
tuple T{
   int a;
   int b;
}

tuple TT{
   T a;
   int b;
}

tuple TTT{
   TT a;
   int b;
}

{T} S0 = ...;
{TT} S1 with a in S0 = ...;
{TT} S2 with a in S1 = ...;
{TT} S3 = < t | t in S2 : t.a.a == 0}; // slicing on S2 is not allowed.
```

.NET interfaces

Renaming in .NET interfaces

All methods with the first character in lowercase have been renamed with the same name, and with the first character in uppercase.

Properties replaced by methods

In `ILOG.CPLEX.Cplex`, the following properties have been replaced by methods.

`ParameterSet` (getter)	—>	`GetParameterSet()`
`ParameterSet` (setter)	—>	`SetParameterSet()`
`Objective`	—>	`GetObjective()`
`DeleteMode` (getter)	—>	`GetDeleteMode()`
`DeleteMode` (setter)	—>	`SetDeleteMode()`
`Algorithm`	—>	`GetAlgorithm()`
`IIS`	—>	`GetIIS()`

Methods replaced by properties

In `ILOG.OPL.OplElement`:

`isExternalData()`	—>	`IsExternalData`

In `ILOG.OPL.OplModel`:

`isInternalData()`	—>	`IsInternalData`
`getName()`	—>	`Name`
`getSourceName()`	—>	`SourceName`

In `ILOG.OPL.OplmodelDefinition`:

`isNull()`	—>	`IsExternalData`

In `ILOG.OPL.OplRunConfiguration`:

`setSettings()`	—>	`Settings`
`getSettings()`	—>	`Settings`
`getCP()`	—>	`CP`

In `ILOG.OPL.OplSettings`:

`isSkipAssert()`	—>	`IsSkipAssert`
`isSkipWarnNeverUsedElements()`	—>	`IsSkipWarnNeverUsedElements`
`getResolverPath()`	—>	`ResolverPath`
`setResolverPath()`	—>	`ResolverPath`
`getTmpDir()`	—>	`TmpDir`
`setTmpDir()`	—>	`TmpDir`
`isKeepTmpFiles()`	—>	`IsKeepTmpFiles`
`getRelaxationLevel()`	—>	`RelaxationLevel`
`setRelaxationLevel()`	—>	`RelaxationLevel`

Databases

Microsoft Access

For Microsoft Access databases (`DBConnection("access",....)`), OPL 6.0 supports both Access 2003 and Microsoft Office Access 2007 through the Office 2007 drivers.

If you obtain an error message `[Microsoft][ODBC Microsoft Access Driver] Cannot open database '(unknown)'.` when using `DBConnection("access",...)` in your `.dat` file, you need to install the Office 2007 drivers on your machine even if you are using Access 2003 databases. You can download these drivers from the: *Microsoft website*.

Oracle

OPL now supports Oracle 10g and Oracle 11g with native drivers. The connection strings are, respectively:

```
DBConnection db("oracle10","<user>/<password>@<DB>");
```

```
DBConnection db("oracle11","<user>/<password>@<DB>");
```

oplrun

- You can start the `oplrun` line command from the **Run** menu in the standard toolbar of the IDE.

  Run>External Tools>Run As>oplrun

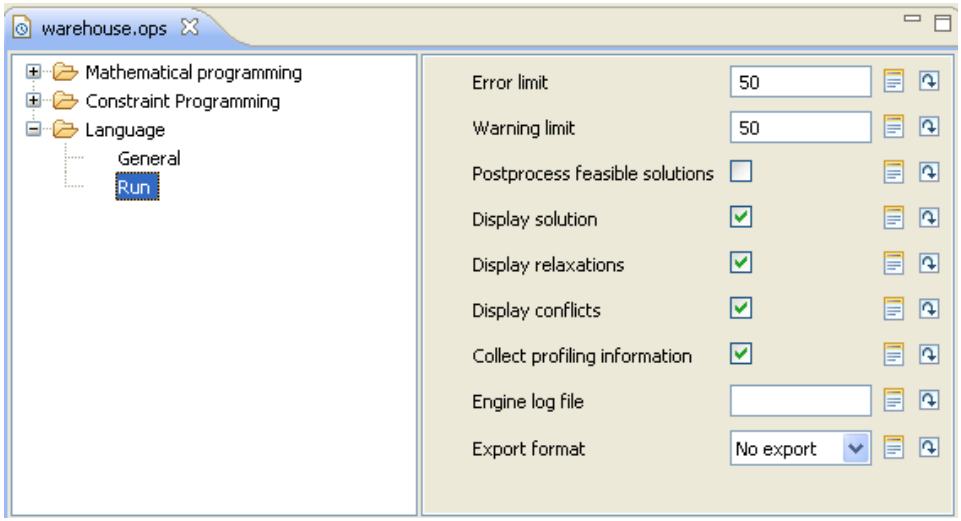
- There is a new flag `-locale`. For information, see the document *oplrun Command Line Interface*

- The flags `-di, -de` have been enhanced. They now support **—** as filename for output to `stdout`. For example:

  ```
  oplrun -di - -p ..\..\..\examples\opl\warehouse
  ```

- `.ops` files support the new settings category **Run** .

- The parameters on the **Run** page of the settings file are no longer restricted to the OPL IDE. They are now also available for `oplrun` and ODM.



Examples

There are new examples to illustrate the scheduling feature. They can be found in the examples directory.

- `<OPL_dir>\examples\opl\sched_*`
- `<OPL_dir>\examples\opl\models\sched_*`

There is also a new example of how to use a CPLEX deterministic model to handle a stochastic problem. It can be found in the examples directory.

`<OPL_dir>\examples\opl\models\YieldStochastic`

Port changes

For a complete list of machine types and library formats (including version numbers of compilers and JDKs) available in the current release, see the file<OPL_dir>/mptable.html distributed with the product.

Summary of port changes

Port Name	Platform	Status	Comment
New in this release			
x86_windows_vs2008 x64_windows_vs2008	Microsoft Visual Studio 2008	New ports available	
x86_debian4.0_4.1 x86-64_debian4.0_4.1	Redhat Enterprise Linux 5.0 Debian 4.0 Suse Linux Enterprise Server 10.0	New ports available	
Deprecated — will not be delivered in the next release			
power32_aix5.2_7.0 power64_aix5.2_7.0	AIX 5.2	Will not be available in the next release	Recommended replacement: AIX 5.3 or a higher version
No longer in the release			
x86-64_rhel4.0_3.4 x86_rhel4.0_3.4	Redhat Enterprise Linux 4.0	Removed. No longer available.	Recommended replacement: x86_debian4.0_4.1 or x86-64_debian4. 0_4.1
x86_.net2003*	Microsoft Visual Studio .NET 2003	Removed. No longer available.	Recommended replacement: Microsoft Visual Studio 2005 or Microsoft Visual Studio 2008

New ports

The following new ports are available with this release:

- New ports, suitable for x86 and x86-64 processors and built on the Debian 4.0 distribution of the Linux operating system, are available in this release. They are compatible with Red Hat Enterprise Linux 5 and Suse Linux Enterprise Server 10.
- New ports supporting Sun UltraSPARC processors and Sun Studio 9 or higher compilers and development environment are available for 32–bit and for 64–bit architectures.
- A new port compatible with Microsoft Visual Studio 2008 is available.

Deprecated and removed ports

- **IBM Power AIX 5**

  The ports named `power32_aix5.2_7.0` and `power64_aix5.2_7.0` are deprecated. IBM has announced plans to stop support for AIX 5.2 in the next twelve months. The recommended replacement is to migrate to AIX 5.3 or a higher version, which is also supported by this port.

- **Red Hat Enterprise Linux 4**

  Ports based on Red Hat Enterprise Linux version 4.0 are deprecated and have been removed. The recommended replacements, named `x86_debian4.0_4.1` and `x86 -64_debian4. 0_4.1`, are compatible with Red Hat Enterprise Linux version 5, Suse Linux Enterprise Server version 10, Debian version 4.

- **Microsoft Visual Studio 2003**

  The ports based on Visual Studio 2003 have been removed, as they are no longer supported by Microsoft. Recommended replacements are based on Microsoft Visual Studio 2008.

New and changed in OPL 5.5

These release notes highlight improvements and new features in OPL 5.5. Please review these notes before using OPL 5.5.

In this section

Deprecated
Lists software that is deprecated in version 5.5 and will not be supported in the next version of OPL.

CPLEX
ILOG OPL Development Studio 5.5 is based on CPLEX 11.0 and CP Optimizer 1.1. This topic lists new and changed features in CPLEX 11.0.

IDE
Lists new and changed features in the OPL IDE.

Language
Describes changes to ordered sets and `assert`statements.

Scripting
Lists the improvements in OPL scripting.

oplrun
The `oplrun` command is now available from the toolbar in the IDE.

Interfaces
Lists improvements to the interfaces to OPL.

Examples and model library

The directory structure of the distributed examples has been reorganized.

Deprecated

Links to other documentation items

Note: The release notes for versions earlier than the current one remain accessible from the table of contents. However, most of the links they provide to the documentation are unclickable textual links, **not** hypertext links, because the structure of the documentation set partly changes from one release to the next.

JDK 1.4

JDK 1.4, also known as J2SE 1.4, has been deprecated. That is, this development kit will no longer be supported after ILOG OPL Development Studio 5.5. Instead, ILOG urges customers who currently use this deprecated development kit to migrate to a more recent version, such as JDK 6, as recommended by Sun Microsystems.

Visual Studio .NET2003

Visual Studio .NET2003 has been deprecated. That is, this compiler version will no longer be supported after ILOG OPL Development Studio 5.5. Instead, ILOG urges customers who currently use this deprecated compiler to migrate to a more recent version, such as Visual Studio .NET2005.

API

The ILOG Concert `setCoef` method has been removed. Use the `setLinearCoef` method instead. This method throws an exception if the invoking expression contains a nonlinear part.

The ILOG Script method `IloOplModel.endAll` is deprecated. See *Improved sharing of data between models* of these Release Notes.

CPLEX

Solution pool

Nonoptimal solutions found during a MIP search are now collected and can be browsed after the search. You can find several solutions without having to run the modified search several times. You can control what solutions enter the pool by using CPLEX parameters and adding specific filters. The feature is also available from flow control scripting. See *Working with the solution pool* in *IDE Tutorials*.

Tuning tool

From one or several run configurations, this tool performs some test runs and suggests some parameter combinations that work better than the default ones. The tool can be invoked from the IDE or from the oplrun command line. See *Using the performance tuning tool* in *IDE Tutorials*.

Deterministic parallel optimizer and dynamic search

OPL fully supports CPLEX deterministic parallel and dynamic search.

IDE

Standard toolbar

The standard toolbar now has two new buttons:

- the **Launch oplrun** button, which enables you to start the oplrun command line from the IDE. See *oplrun* in these release notes.
- the **Tune model** button, which enables you to start the tuning process. See *Tuning tool* in these release notes.

Problem Browser

At the top of the **Problem Browser**, a drop-down list displays the solutions collected in the solution pool, if any, in addition to the final solution. See *Solution pool* in these release notes.

Math programming options

More CPLEX parameters are available as OPL settings. See Setting MP options.

Language

Ordered sets

OPL now supports ordered sets and sorted sets. While default sets keep the construction order of the elements, sorted sets use the natural order. You can use them when the construction order is not relevant. See *Sorted and ordered sets* in the *Language Reference Manual*.

Labeled asserts

By labeling the constraints used in assert statements, you can now customize the way failed assertions are handled. By default, OPL now gives more detailed feedback on which part of the assertion has failed. Finding data errors is now easier. See *Labeled assert statements* in the *Language Reference Manual*.

Scripting

The setCoef methods

The methods `IloConstraint.setCoef` and `IloObjective.setCoef` have been fixed in ILOG Concert. They are now allowed only when the expression inside the range or objective is a simple real linear expression (like `x+y`). These methods do not work for aggregate sums. For aggregate sums, you have to use the methods directly on the CPLEX instance:

```
cplex.setObjCoef(dvar, coef)
```

and

```
cplex.setCoef(ct, dvar, coef)
```

These instructions directly change the engine representation without going through Concert.

Ending intermediate elements

You can now end the OPL elements you don’t need in any scripting context, not only in flow control. This allows you to reduce overall memory consumption by ending intermediate elements. See *A few tips* in the *Language User’s Manual* for general information on writing scripts.

Improved sharing of data between models

In flow control scripting, OPL memory management has been improved, so that the `end` method now frees as much memory as possible, just as the method `endAll` used to do so far. Therefore, you can now share data across model instances and control memory consumption efficiently using the method `IloOplModel.end`. As a consequence, the method `endAll` is now deprecated.

OPL functions in script statements

You can use most OPL modeling functions, especially to handle sets, within ILOG Script statements if you introduce them with the `Opl` namespace. As a result, you can carry out more data processing; for example, accessing functions such as `item` or `ord`. See OPL functions in the *Language Quick Reference*.

oplrun

You can start the oplrun command line from the **Launch oplrun** button in the standard toolbar of the IDE. You can also configure from the IDE how the oplrun command line is launched so that it is easy to test a model in deployment mode (for example, in 64-bit mode or without the IDE memory overhead). See Using the command line interface in *oplrun Reference*.

Warnings are now turned off by default. You can turn then on by adding the `-w` option in the command line. The `-Xw` option does not exist any more.

Interfaces

CPLEX callbacks in Java and .NET

CPLEX callbacks are now supported with parallel search in Java and .NET interfaces.

Postprocessing of feasible solutions

A custom CPLEX callback can now call postprocessing instructions on solutions.

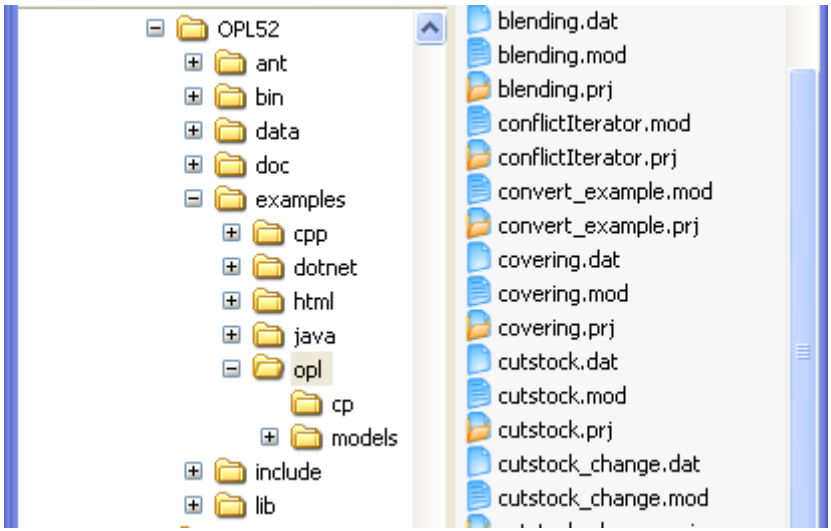
Exceptions in .NET

A bug has been fixed in ILOG Concert .NET interfaces. Now all exceptions thrown by OPL inherit from `ILOG.Concert.Exception`. So it is no longer possible to catch/throw these exceptions as `java.lang.Exception`.

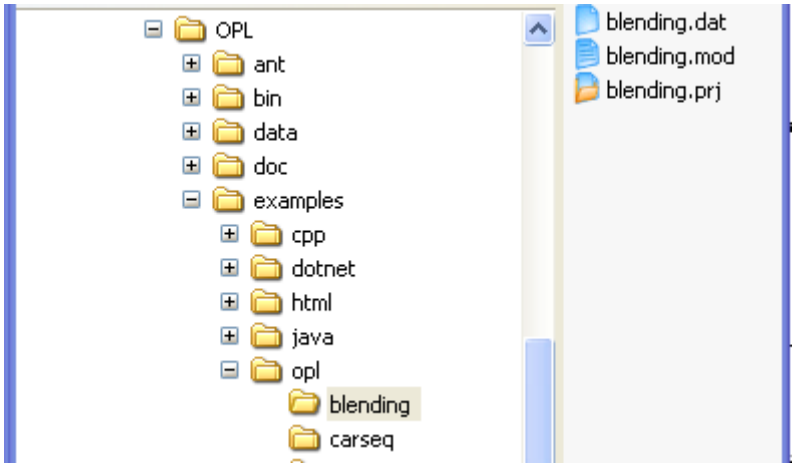
Examples and model library

The structure of the examples delivered with the product distribution has been reorganized.

- All the project, model, data, and settings files for a given example are now located in a single parent directory with the same name. For example, for the blending code sample, file `blending.dat`, `blending.mod`, and `blending.prj` are included in the `blending` directory. Compare *Old directory structure for OPL distributed examples* and *New directory structure for OPL distributed examples*.



Old directory structure for OPL distributed examples



New directory structure for OPL distributed examples

- The `<OPL_dir>\examples\opl\cp` directory no longer exists. The subdirectories it contained for constraint-programming examples are now directly under `<OPL_dir>\examples\opl`. For example, the path to the car sequencing example is now:

  `<OPL_dir>\examples\opl\carseq\carseq.prj`

  instead of

  `<OPL_dir>\examples\opl\cp\carseq.prj`

- To better apply the concept of "project" that underlies the OPL design, files pertaining to similar MP problems have been attached to a single `.prj` file. As a consequence, some examples that were projects of their own are now run configurations within other existing projects. Each run configuration has been named so as to highlight the specific aspect of its model. However, no model or data files have been removed in the process. *Code samples: projects turned into run configurations* summarizes the resulting changes. See also the OPL examples that use CPLEX (Mathematical Programming) in the *Samples* manual.

Code samples: projects turned into run configurations

This project no longer exists.	It has become this run configuration...	...of this project
cutstock_change.prj	Alternation formulation fixed number of patterns	**cutstock.prj**
cutstock_ext_main.prj	Solve using external call	
cutstock_int_main.prj	Solve full integer problem using column generation	
cutstock_main.prj	Solve the full relaxed problem using column generation	
cutstock-sub.prj	Submodel used in the main scripts	
gas1.prj	Without dat file	**gas.prj**
gasn.prj	Using indices in the arrays	
knapsackp.prj	Solving using CPLEX priorities	**knapsack.prj**
mulprod_change_main.prj	Modify the CPLEX model incrementally	**mulprod.prj**
mulprod_main.prj	Solve a sequence of modified OPL models by changing data	
oilDB.prj	Loads data from a database	**oil.prj**
oilDB2.prj	Better database reading to initialize data	
oilSheet.prj	Loads data from a spreadsheet	
product.prj	Using tuples	**production.prj**
productn.prj	Using named data	
sailcopw.prj	Piecewise linear cost	**sailco.prj**
sailcopwg1.prj	Generalized piecewise linear model - 2 pieces	
sailcopwg2.prj	Generalized piecewise linear model - 3 pieces	
sailcopwg3.prj	Generalized piecewise linear model - 4 pieces	
transp1.prj	Configuration1	**transp.prj**
transp2.prj	Using sparsity	
transp3.prj	Using better sparsity	
transp4.prj	Using even better sparsity	
scalableWarehouse.prj	With scalable data	**warehouse.prj**

New and changed in OPL 5.2

These release notes highlight improvements and new features in OPL 5.2. Please review these notes before using OPL 5.2. The main change in OPL 5.2 over OPL 5.1 is the introduction of constraint programming features.

In this section

Support for constraint programming
The CP Optimizer engine supports: syntax for specialized constraints, custom propagators, labeled constraints, and ODM application generation.

Deprecated
Lists software that is deprecated in 5.2 and will not be supported in the next version of OPL.

Licensing - trial version
Describes the licensing for the OPL-CPLEX Analyst Studio online trial version.

Performance
Describes improved performance on Unix and Windows platforms.

Localization
Describes improvements in the localization and internationalization of OPL.

Terminology
Mentions a change in terminology.

Changes to the language
Lists the changes to the OPL language.

Changes to the IDE

Lists the changes to the OPL IDE.

Examples and models

New code samples and industry models have been added to illustrate how to model constraint programming problems with OPL.

Changes to the documentation set

The table of contents reflects a few changes to Structure and Contents.

Support for constraint programming

Older versions of OPL (up to version 3.7) supported constraint programming for solving models. In OPL 4.0, however, because of a complete rewrite and redesign of the product, support for constraint programming was temporarily removed.

With this release of ILOG OPL Development Studio, support for core constraint programming featuring, in particular, a powerful default search algorithm is being re-introduced.

This new CP Optimizer engine supports logical constraints, nonlinear constraints, compatibility constraints, specialized constraints through keywords such as `allDifferent`, `count`, `lex`, `pack`, `inverse`, and more.

You can write a complete model to be solved using the CP Optimizer engine, edit it and solve it from the OPL IDE, from the oplrun command line tool, and from the APIs. From the OPL IDE, you can also browse the solution and look at the engine log, progress, and statistics.

The CP Optimizer engine supports:

- Syntax for specialized constraints
- Custom propagators
- Labeled constraints
- ODM application generation

Important: OPL Development Studio 5.2 does not support constraint-programming-based scheduling. If you have used an earlier version of OPL Studio to write scheduling applications based on constraint programming, please send email to *oplcp@ilog.com* for information about support for CP-based scheduling in a future version of ILOG OPL Development Studio. ILOG Scheduler features such as Activity, DiscreteResource, the precedes constraint, the requires constraint are not supported by OPL Development Studio 5.2.

The constraint programming features of ILOG OPL Development Studio 5.2 are documented in several places of the documentation set.

Read:

- *Migration from OPL 3.x (CP projects)* to learn how to change the syntax of CP statements in models originally written in OPL 3.7.x
- the new keywords and functions in the *Language Quick Reference*
- a short tour of OPL constraint programming in *Constraint programming* and *Constraint programming versus mathematical programming* in the *Language User's Manual*
- a typical CP problem in *Constraint programming: an inventory matching problem* in the *Language User's Manual*
- more CP application areas described in *Applications of constraint programming* in the *Language User's Manual*

- a presentation of CP constraints in *Constraints available in constraint programming* in the *Language Reference Manual*
- how to use script statements in CP models in *Using IBM ILOG Script in constraint programming* in the *Language User's Manual*
- descriptions of the CP Engine Log and Engine Statistics output window in *Understanding solving statistics and progress (CP models)* of *IDE Tutorials*
- a description of the CP settings available from the IDE in Constraint programming options of *IDE Reference*
- the descriptions of the new CP code samples and industry models delivered with OPL in OPL examples that use CP Optimizer (Constraint Programming) and in OPL model library of the *Language and Interfaces Manual*.
- CP-related terms in the Glossary

Deprecated

Links to other doc items

Note: The release notes for versions earlier than the current one remain accessible from the table of contents. However, most of the links they provide to the documentation are unclickable textual links, **not** hypertext links, because the structure of the documentation set partly changes from one release to the next.

.NET Framework

The recommended version of the .NET Framework is 2.0. Version 1.1 is still supported for compatibility reasons but will no longer be supported in a future release.

Database support

Oracle 8i is still supported in this release but will no longer be supported in the next major release.

Deprecated keywords

For deprecated keywords, see *OPL syntax* in these Release Notes.

Deprecated APIs

Deprecated APIs in OPL 5.2

Class	Constructor/Methods
`IloOplDataHandler`	`IloOplDataHandler(IloEnv env, IloOplSettings settings,` `IloOplDataHandler(IloEnv env, ostream& outs);`
`IloOplElementz`	`ILO_OPL_DEPRECATED IloBool isData() const;` `ILO_OPL_DEPRECATED IloBool isCalculated() const;`

`IloOplModel`	`IloBool hasMain()`
	`void printData(ostream& os)`
	`void printCalculatedData(ostream& os)`

`IloOplModelSource`	`IloOplModelSource(IloEnv env, const char* filename, Ilo`

Class	Constructor/Methods
	IloOplModelSource(IloEnv env, istream& ins, const char*
IloOplModelDefinition	IloOplModelDefinition(IloOplModelSource source, IloOplE
IloOplRunConfiguration	void setCompiledModel(IloBool /*compiled*/)
IloOplSettings	void setWithDecisionVariableNames(IloBool with) IloBool isWithDecisionVariableNames() const

Licensing - trial version

OPL-CPLEX Analyst Studio has an online trial version. It is the regular OPL installer, but no license keys are issued.

If OPL finds no license keys, it restricts its usage as follows:

- Only the IDE is available.
- CPLEX is available, but limited to only 500 constraints and 500 variables.
- CP Optimizer is available, but the search space size, before propagation, is restricted to the equivalent of 1000 binary (0-1) variables.

Performance

UNIX deployment ports are 25% faster, Windows deployment ports are a little faster, depending on the configuration.

Localization

The IDE is now localized in German, French, and Spanish.

Because of better internationalization of OPL, some models that work in OPL 5.1 may not work in OPL 5.2 if they are not used with the right locale. Make sure to use the right locale.

Terminology

Partial solutions are now called “feasible solutions”.

Changes to the language

Changes to the OPL language:

- Specific syntax to support the CP Optimizer engine
- Aggregation of the `and` and `or` operators for Boolean expressions
- Aggregation of the `union` and `inter` operators for set expressions
- New functions and keywords: `all`, `allDifferent`, `allMinDistance`, `allowedAssignments`, `and`, `append`, `count`, `CP`, `CPLEX`, `element`, `forbiddenAssignments`, `inverse`, `lex`, `or`, `pack`, `sgn`, `standardDeviation`, `using`
- Additional syntax for the `card` function: `card(range)`
- Tuple arrays can be populated as internal data.
- Transitivity of key properties changed
  - The properties of subtuple keys are now taken into account even if they are not declared in the enclosing tuple.
  - The default rule is: No key declared implies that all components are keys.
- Integer conditions: OPL accepts Boolean expressions in a context of integers. However, it does not accept integers for conditions. Because of a bug, OPL accepted conditional assignments with integer conditions:

  ```
  (i) ? 1 : 0
  ```

  This has been fixed in OPL 5.2. Now you need to write:

  ```
  (i>0) ? 1 : 0
  ```

  to get the same result.
- Database connection: a new implementation replaces the current ODBC access to Access database by transparently locate the Access mdb file. See DBConnection in the *Language Quick Reference* and *Connection to a database* in *IDE Tutorials*.
- Processing order: `assert` and `execute` statements are now processed in declaration order.

Changes to the IDE

Issues output window

Pressing F1 on a line in the Issues output window now opens the documentation of error messages.

Problem Browser

Enhanced display of the problem structure.

Outline

The **Model Outline** indicates the number of each element type in the model. See The Outline in *IDE Reference*.

Settings Editor

- New settings are available for constraint programming. See Constraint programming options in *IDE Reference*.
- The **CPLEX Log file** option has been changed to **Engine Log file** and moved to the OPL Language sections.
- A new OPL Language setting, **Relaxation Level**, is available to control the way automatic relaxations are done. Variables can be skipped from the list of candidates to be relaxed. See *Relaxing infeasible models* in *IDE Tutorials*.

Output window

- Pressing F1 on a line in the Issues output window now opens the documentation of error messages.
- The **Profiler** displays the **Peak Memory** and **Local Memory** data. See *Profiling the execution of a model* in *IDE Tutorials*.

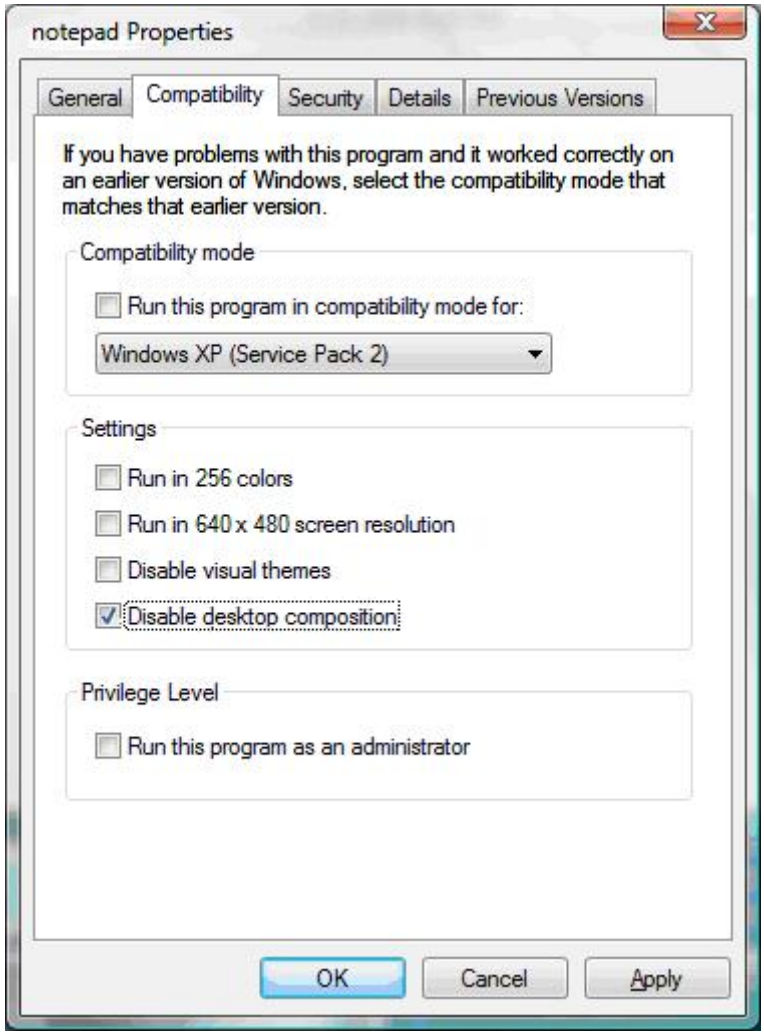
Vista graphical issues (Aero only)

Microsoft Windows Vista Aero does not support XOR drawing to the background window. When dragging dockable windows or drag/dropping items to the Projects window, the background is erased.

The workaround consists in activating a compatibility flag.

To activate a compatibility flag:

1. In the Windows Explorer, right-click the OPL IDE executable and select Properties.
2. Click the Compatibility tab, then check the option **Disable desktop composition**.


notepad Properties
General
Compatibility
Security
Details
Previous Versions
If you have problems with this program and it worked correctly on an earlier version of Windows, select the compatibility mode that matches that earlier version.
Compatibility mode
Run this program in compatibility mode for:
Windows XP (Service Pack 2)
Settings
Run in 256 colors
Run in 640 x 480 screen resolution
Disable visual themes
Disable desktop composition
Privilege Level
Run this program as an administrator
OK
Cancel
Apply

Examples and models

Where to find the files

You can find the corresponding files at the following location:

`<OPL_dir>\examples\opl\cp` for CP code samples

`<OPL_dir>\examples\opl\models` for CP industry and academic models

where `<OPL_dir>` is the OPL installation directory.

See the *Language and Interfaces Samples* manual for detailed descriptions.

Code samples

Code samples for the following CP problem or modeling features have been added to the `examples\opl\cp` directory:

Audition scheduling: `talent.prj`

Car sequencing: `carseq.prj`

Color matching: `color.prj`

Column generation: `vellino.prj` (multi-model project)

Configuration: `config.prj`

Euler problem: `euler.prj`

Float expressions: `floatexpr.prj`

Inventory matching; `steelmill.prj`

Progressive party problem: `PPP.prj`

Quadratic assignment: `quadassign.prj`

School class scheduling: `timetabling.prj`

Sports game and team scheduling: `sports.prj`

Team building: `teambuilding.prj`

See OPL examples that use CP Optimizer (Constraint Programming) in the *Language and Interfaces Manual* for details.

Industry and academic models

All the models distributed in the Model Library directory

`<OPL_dir>\examples\opl\models`

have been rewritten as necessary to follow the recommended modeling practices:

Namely:

- decision expressions (`dexpr` keyword) for objectives
- labeling of constraints (rather than names or nothing at all),
- array initialization instead of scripting statements whenever possible

See OPL model library in the *Language and Interfaces Manual* for details.

Changes to the documentation set

Structure

- Information on how to find your way through the documentation set is now an entry in the Table of Contents.
- The Quick Start Guide to OPL and the reference information on the IDE GUI, which were formerly included in the *Getting Started with the IDE*, now appear as separate entries in the table of contents: see *Quick Start to OPL* and *IDE Reference*.
- A Tutorials group has been created in the table of contents to give users better access to hands-on exercises and industry problems in IDE Getting Started and Tutorials, and Language Tutorials.
- Decision variables and decision expressions are now documented in a separate chapter of the *Language Reference Manual*: *Decision types*.
- The Reference Manual for OPL Extensions to ILOG Script can now be found in the Language group of the table of contents.

Content

The new content mostly concerns Constraint Programming. See *Support for constraint programming*.

New and changed in OPL 5.1

OPL 5.1 is a minor update to OPL 5.0. In this release, you will benefit from more ports, improved performance, external function calls, table loading, `dexpr` arrays, scripting in data files, better usability in the profiler, bug report packaging, localization, and more.

In this section

Deprecated features
Lists software that is deprecated in version 5.1 and will not be supported in the next version of OPL.

Performance and new ports
Describes performance improvements in OPL 5.1 and lists the new ports available.

Changes to the language
Describes the language changes in OPL and ILOG Script.

Changes to the IDE
Lists the changes made in the user interface.

oplrun
`oplrun` has been improved.

Miscellaneous
Resolution of file paths is more systematic. Compiled model files are detected automatically.

Incompatibilities
Two incompatibilities with the previous version, OPL 5.0.

Deprecated features

Links to other documentation items

Note: The release notes for versions earlier than the current one remain accessible from the table of contents. However, most of the links they provide to the documentation are unclickable textual links, **not** hypertext links, because the structure of the documentation set partly changes from one release to the next.

OPL syntax

The current OPL keywords for database connection are still supported but are planned to be deprecated in the next major release. It is recommended that you start using the modified syntax (with upper case letters) in this OPL version.

Recommended DB connection keywords

Deprecated	Use this keyword instead
DBconnection	DBConnection
DBread	DBRead
DBupdate	DBUpdate
DBexecute	DBExecute

OPL interfaces

The following methods are deprecated.

Deprecated interfaces in 5.1

Classes	Method Signatures

`IloOplModel`	`hasMain()`
	`printData(ostream&)`
	`printCalculatedData(ostream&)`

`IloOplModelSource`	`IloOplModelSource(IloEnv,const char*,IloBool)` `IloOplModelSource(IloEnv,istream&,const char*,IloBool)`
`IloOplModelDefinition`	`IloOplModelDefinition(IloOplModelSource,IloOplErrorHandler)`
`IloOplRunConfiguration`	`setCompiledModel(IloBool)`
`IloOplSettings`	`setWithDecisionVariableNames()`

Classes	Method Signatures
	`WithDecisionVariableNames()`
`IloOplElement`	`isData()` `isCalculatedData()`

Performance and new ports

Performance improvements

OPL 5.1 is around 10% faster and uses 10% less memory than OPL 5.0 on our benchmark suite.

New ports

OPL is now available for Microsoft Visual Studio 2005, for both 32-bit and 64-bit versions.

However, the 64-bit version has the following limitations:

- No IDE.
- C++ and Java interfaces are available, but .NET interfaces are not. Moreover, Java interfaces work only with “Windows AMD64 Platform - J2SE(TM) Development Kit 5.0”.
- Microsoft Excel supports only 32-bit input and output.

Changes to the language

External function calls

You can now extend scripting blocks in OPL models with external functionality implemented in Java. Among many other use cases, it makes it possible to combine the use of the CPLEX algorithm with another algorithm written in Java in a more complex optimization process. See *Tutorial: External functions* in the *Language User's Manual*.

Arrays of decision expressions

After decision expressions (`dexpr`) were introduced in OPL 5.0, decision expression arrays are now supported as well. See *Decision Variables and Expressions* in the *Language Reference Manual*.

Table loading

OPL models can now read columns from any table into an OPL array using `DBread` statements. See *Working with external data* in the *IDE Tutorials and Customization* manual.

Script statements in data files

You can now write script functions in a `.dat` file to define custom ways of reading and formatting data. The `invoke` and `prepare` OPL keywords have been created to support this functionality. See *Using IBM ILOG Script in data files* in the *Language User's Manual*.

Localization

The OPL IDE is now localized in Japanese and Simplified Chinese. See Internationalization.

Changes to the IDE

The changes in the OPL IDE affect:

- *The Projects window*
- *The Problem Browser*
- *The Profiler*
- *Bug report packaging*
- *oplrun*
- *Miscellaneous*

See *Getting Started with the IDE*, especially the *Tour of the OPL Graphical User Interface*, and the *IDE User's Manual* for more information.

Tooltips

In addition to the standard button tooltips, click the **Help** button to read quick reference information on some elements of the main window.

The Projects window

Usability has been improved: you can drag and drop files and rename folders. Run configurations are updated accordingly.

The Problem Browser

The **Problem Browser** now provides a more user-friendly display of tuple sets and one-dimensional tuple maps, thus giving a better overview of the tuple structure.

Output tabs

The **CPLEX Log** and **CPLEX Statistics** output tabs have been renamed to **Engine Log** and **Engine Statistics** to become engine-independent.

The Profiler

The **Profiler** now includes more detailed information and has a better look-and-feel for better usability. See *Profiling the execution of a model* in the *IDE Tutorials and Customization* manual.

Content

- Extraction of the model: details are displayed for each labeled constraint. This allows you to find out which constraint consumes more memory or takes more time to be created.

- CPLEX search: the tree search now provides more information on merged time for the different kinds of actions.

Usability

You can now easily highlight significant figures using memory and time thresholds. You can also order columns by clicking their headers.

Bug report packaging

A new wizard is available from the Help menu for you to create an archive of your run configuration in a few clicks and send it to ILOG Customer Support for troubleshooting purpose. However, this facility does not cover database connectivity issues. See *Packaging a bug report* in *Getting Started with the IDE*.

oplrun

It is now possible to override values of simple-type data elements from the command line. You will find it useful to use oplrun as a testing environment and run the same model repeatedly with slightly different input.

Miscellaneous

- Resolution of file paths is now more systematic: included models and scripts, or other files such as input and output files in scripting statements, are now found at a clearly documented place. See *Path for resolving resources* in the *IDE Tutorials and Customization* manual.
- Compiled model files are now detected automatically.

Incompatibilities

- Model files compiled with OPL 5.0 or earlier need to be recompiled.
- Relative paths in scripting statements for `IloOplInputFile` and `IloOplOutputFile` instances are now resolved relative to the invoking model file, not relative to the current working directory.

New and changed in OPL 5.0

OPL 5.0 is a major update to OPL 4.x. It is also built upon ILOG Concert Technology. In addition to significant gains of performance, OPL 5.0 brings the changes listed in these release notes.

In this section

Deprecated features
Lists software that is deprecated in 5.0 and will not be supported in the next version of OPL.

Support for ILOG ODM
Introduces ILOG Optimization Decision Manager (ODM).

Changes to the language
Discusses changes to the OPL language and their impact.

Changes to the IDE
Describes the changes that affect the main window, and project management and execution.

Changes to the documentation
Describes changes to the structure of the OPL documentation set.

Deprecated features

Constraint names are deprecated in favor of constraint labels. See *Constraint labels*.

Note: The release notes for versions earlier than the current one remain accessible from the table of contents. However, most of the links they provide to the documentation are unclickable textual links, **not** hypertext links, because the structure of the documentation set partly changes from one release to the next.

Support for ILOG ODM

Models developed in ILOG OPL Development Studio 5.0 can be used directly with ILOG ODM 2.0 to create applications. See *From Problem Solving to What-If Scenarios: A Presentation of ILOG Optimization Decision Manager* in the *Getting Started with the IDE*.

A specific `dexpr` syntax has been introduced to help you write objectives more efficiently when working with ODM. For example:

```
dexpr int allocationCost = sum(i in r) allocation[i];
```

See the ODM documentation for more information.

Note that arrays of decision expressions (`dexpr` arrays) are not supported in OPL 5.0. They will be added in the next minor release.

Changes to the language

Discusses changes to the OPL language and their impact.

In this section

Changes to the modeling language
Describes changes affecting expressions, tuple keys, data consistency, constraint labels, logical constraints and expressions, compatibility.

ILOG Script
Describes changes to ILOG Script: incrementality, control of the CPLEX warm start and basis, control of relaxation and conflicts, tuple sets

oplrun
The `oplrun` command can be used to execute OPL projects.

Changes to the modeling language

Expressions

See *Support for ILOG ODM*.

Tuple keys

As in databases, some fields can be set as keys. Using keys in tuple declarations enables you to search, display, etc. on tuples just by using the keys as unique identifiers. In particular, slicing is more efficient, and sets use less memory. For example:

```
tuple nurse {
  key string name;
  int seniority;
  int qualification;
  int payRate;
}
```

You can access arrays indexed by tuple sets of nurses by using their names only. Indexing tuples display as key values only.

See the **nurses example** and *Keys in tuple declaration* in the *Language Reference Manual*.

Data consistency

You can use the keyword `with` to ensure that a given part of a tuple belongs to a given set. This syntax is recommended because it saves memory. For example:

```
{int} Seniorities = {1, 3, 5};
 {nurse} Nurses with seniority in Seniorities = ...;
```

See *Data consistency* in the *Language Reference Manual*.

Constraint labels

Constraint names are deprecated (although still supported) and replaced by a more powerful and flexible labeling feature which does not require that you declare the constraint first. For example:

```
forall(i in r)
  ct: x[i] <= a[i];
```

See the **volsay.mod example** and *Constraint labels* in the *Language Reference Manual*.

Logical constraints and expressions

As in CPLEX, you can now use logical constraints and expressions inside an OPL model solved by CPLEX.

Example:

```
forall(i in r) (x[i]<=5) || (x[i]>=10);
```

See the **food manufact example** and *Tutorial: Using CPLEX logical constraints* in the *Language User's Manual*.

Compatibility

Range expressions

Range expressions must have ground boundaries. For example, the following model is not allowed any more in OPL 5.0, whereas it was in OPL 4.2.

```
dvar int x in 0..5;
dvar int y;
minimize x;
subject to {
   0<=y<=x;
}
```

In OPL 5.0, you must write:

```
{
dvar int x in 0..5;
dvar int y;
minimize x;
subject to {
   0<=y;
   y<=x;
}
```

ILOG Script

Script incrementality

Specific methods of ILOG Script extension classes for OPL allow incremental modification of the CPLEX model generated by OPL. For example:

```
thisOplModel.ct1["flour"][t].UB = capFlour;
```

See the **mulprod_change_main example** and *Modifying the CPLEX matrix incrementally* in the *Language User's Manual*.

Control of the CPLEX warm start and basis

Specific methods of the `IloOplCplexBasis` and `IloOplCplexVectors` classes enable you to control for what part of the model you want to save or restore the basis or vectors.

See the **warmstart example** and *Setting an initial solution for the CPLEX engine* in the *Language User's Manual*.

Control of relaxation and conflicts

Specific methods of the `IloOplRelaxationIterator` and `IloOplConflictIterator` classes enable you to set weights and priorities on constraints to specify which part of the model to consider and how.

See the **conflictIterator example** and *Setting preferences on the search for conflicts and relaxations* in the *Language User's Manual*.

Projects and run configurations

New classes, both in the Interfaces and ILOG Script libraries, make it much easier to create instances of `IloOplModel` from projects or run configurations.

See:

- the **cutstock_int_main.mod example**
- *Using run configuration and project instances* in the *Language User's Manual*.
- *Using run configuration and projects* in the *Integration Libraries User's Manual*.

Tuple sets

The new scripting methods `get` and `contains` are available on `IloTupleSet` objects. Use the method `get` rather than `find` to retrieve errors.

oplrun

You can now use the `oplrun` command to execute projects directly (instead of only models so far).

See oplrun Command Line Interface.

Changes to the IDE

Describes the changes that affect the main window, and project management and execution.

In this section

The Main window
Describes changes to the menu bar, the toolbars, and the panes in the Main window.

Project management and execution
Introduces settings files and run configurations.

The Main window

See *The Main Window* in *Getting Started with the IDE* for more information.

The menu bar

The menu bar contains fewer menus: **File**, **Edit**, **View**, **Execution**, **Window**, **Help**.

IDE menus, changes from 4.2 to 5.0 presents what menu and commands have removed, added, renamed, or moved.

IDE menus, changes from 4.2 to 5.0

4.2 Menu	Removed	Added	Renamed
File	Generate Compiled Model File Generate External Data File Generate Calculated Data File Appear now as buttons in the standard toolbar. See *Generating output files* in *Getting Started with the IDE*	Generate ODM Application Configure ODM Application Deploy ODM Application See *File Menu* in *Getting Started with the IDE*	Close Current Editor to Close Current Document
Edit	Keyboard shortcut for Kill Line	Add/Remove Breakpoint	
View		Show/Hide Outline	Show/Hide Model Browser to Show/Hide Problem Browser
Project	Menu removed Commands moved to right-click contextual menu on a project See *Right-Click Commands* in *Getting Started with the IDE*		
Execution		Debug	Browse Active Model to Browse
Debug	Menu removed		
Options	Menu removed Project Options command		General Settings to Preferences, moved to Window menu
Window		Preferences	

4.2 Menu	Removed	Added	Renamed
		See *IDE Preferences*in *IDE Tutorials and Customization* manual	
Help		ODM Contents	

The toolbars

The former standard toolbar has been split and the commands that depend on the selected file type appear in the Document toolbar.

For quicker interaction, the following **Browse**, **Run**, **Debug**, and **Generate External/ Calculated Data File** buttons are associated with a small down-pointing arrow which you can click to directly select the file to which you want to apply the corresponding action.

See *Toolbars* in *Getting Started with the IDE*.

The Start window

In addition to quick access to recent files, the Start window contains shortcuts to the OPL web site, to opening and creating projects, and to clearing the list of recent files. See *The Start Window* in *Getting Started with the IDE*.

The Projects window

Has a toolbar with buttons for actions on projects, project documents, and run configurations. See *The Projects Window* in *Getting Started with the IDE*.

The Outline

The new **Outline** window displays an alphabetically-sorted tree representation of the model or settings file you select within a project, for easier navigation across its structure. The model or settings outline is updated dynamically as you make changes. When you edit settings in an `.ops` file, the **Outline** shows all the settings that are different from their original value, as well as the new values. See *The Outline* in *Getting Started with the IDE*.

The Model Browser

This window has been renamed to **Problem Browser** because it reflects the structure of the problem instantiated in the active model whereas the **Outline** represents the structure of the selected file in the Editing area. Specifically, the **Model Outline** displayed when a `.mod` file is selected reflects the model itself (as an expression of a problem) in the order of declaration. See *Understanding the Problem Browser* in *Getting Started with the IDE*.

The Output window

Issues tab

Errors and messages are displayed dynamically, in a more readable manner, in the **Issues** tab as you edit the model. From the Issues tab, you access more information about each error message.

CPLEX Statistics

Redesigned and more complete. See *Understanding solving statistics and progress (MP models)* in the *IDE Tutorials and Customization* manual.

Conflicts and Relaxations

Redesigned and more complete. See *Relaxing infeasible models* in the *IDE Tutorials and Customization* manual.

Profiler

The profiler now includes information on the CPLEX search. The proportions of time spent in the presolve, cuts, etc. are now given.

Project management and execution

Projects are more powerful and can include several models, data sets, and settings files. You create run configurations to define what model, data, and settings are executed at run time.

Creating projects

When you select **File>New>Project** from the menu bar or click the **Create project** button in the **Start** window, the **Create project** dialog box only lets you specify the name, description, and location of the new project and, optionally, whether to immediately create a settings file. You may add data and more settings later from the buttons of the Projects toolbar, after you have closed the dialog box by clicking **Create project** in it. See *Creating a project* in *Getting Started with the IDE*.

You can also use the **Create draft project** button in the **Start** window as a shortcut to quickly sketch a draft model if it is not time yet for a full-fledged project.

Settings files

Settings are not stored in the project files any more, but in separate `.ops` files. Multiple settings files can exist within a project. You can easily add (or remove) settings files to run configurations.

See also *Getting Started with the IDE*, especially the *Tour of the OPL Graphical User Interface*, and the *IDE Tutorials and Customization* manual for more information.

Changes to the documentation

In addition to the new sections and tutorials created to document the changes described in the previous sections, the structure of the documentation set has changed as follows:

- The IDE is documented in two separate manuals:
  - *Getting Started with the IDE*: provides a quick start to OPL, a tutorial to get familiar with the IDE, and a description of the graphical user interface.
  - The *IDE Tutorials and Customization* manual is a set of tutorials to the main features of OPL as used through the IDE, and includes a description of the IDE preferences and of the mathematical programming options available in OPL.
- The **Reference Manual of ILOG Script Extensions for OPL** is accessible from the table of contents (instead of being part of the *Language Reference Manual*) and has a "JavaDoc-like" look-and-feel.
- The `oplrun` command is documented as a separate item for more visibility and provides specific information on running OPL on AIX platforms.
- Error messages: the documentation of error messages is accessible from the Output window of the IDE.
- The way to specify mipsearch has changed between OPL 3 and OPL 4 and later. It is now documented in *Branching (mipsearch)*, in these Release Notes.
- The Bibliography has been moved out of the *Language User's Manual* up to the top level and is now accessible from the table of contents.

Index

Symbols

.NET API
  exceptions **43**

A

aggregate sums, methods to use **41**
AIX platforms **88**
and operator **57**
API **49**
arrays
  of decision expressions
    not supported in 5.0 **75**
    supported in 5.1 **67**
assert, OPL keyword
  processing order of statements **57**

B

Boolean expressions
  aggregation of and/or **57**
bug reports
  packaging for Customer Support **69**
buttons
  Create draft project **87**
  Create project **87**

C

calls to external functions **67**
changes to the language
  from 4.2 to 5.0 **77**
  from 5.0 to 5.1 **67**
  from 5.1 to 5.2 **57**
changes, global
  from 4.2 to 5.0 **74**
  from 5.0 to 5.1 **63**
  from 5.1 to 5.2 **49**
  from 5.2 to 5.5 **37**
Chinese IDE **67**
code samples
  directory structure changed **44**
compatibility **79**
  IDE and Windows Vista **58**
compatibility constraints in CP **49**
compiled model files
  automatic detection **71**
  need recompiling in 5.1 **72**
conflicts
  control by means of ILOG Script **80**
constraint programming
  supported in 5.2 **49**
constraints
  logical **79**
  names changed to labels **78**
  supported by the CP engine **49**
CP Optimizer
  trial mode **53**
CPLEX callbacks **43**
CPLEX deterministic parallel MIP optimization **38**
CPLEX dynamic search **38**
CPLEX engine
  trial mode **53**
  warm start **80**
CPLEX tuning tool **38**
Create draft project, button **87**
Create project, button **87**

D

data
  consistency **78**
  improved sharing between models **41**
data files
  scripting statements in **67**
decision expressions
  arrays of, not supported in 5.0 **75**
  arrays of, supported in 5.1 **67**
deprecated
  in 5.1 **64**
  in 5.2 **51**

© Copyright IBM Corp. 1987, 2009

in 5.5 **37**
dexpr, OPL keyword **75**
Disable desktop composition, Windows Vista option **58**
documentation
changes from 4.2 to 5.0 **88**

E

end method **41**
endAll method **41**
ending objects **41**
error messages **86**
execute, ILOG Script block
processing order **57**
expressions
changes in 5.0 **78**
logical **79**
of decision variables **67**
external function calls **67**

F

feasible solutions
postprocessing **43**
function calls
external **67**

G

ground
boundaries for range expressions **79**

H

Help button **68**

I

IDE
changes in 5.0 **83**
changes in 5.1 **68**
changes in 5.2 **58**
changes in 5.5 **39**
ILOG Scheduler
not supported in 5.2 **49**
ILOG Script
changes in 5.5 **41**
control of relaxation and conflicts **80**
control of warm start **80**
ending objects **41**
in data files **67**
incrementality **80**
OPL modeling functions available **41**
projects and run configurations **80**
IloOplDataHandler class
deprecated constructors **51**
IloOplElement class
deprecated methods **51**
IloOplModel class
deprecated methods **51**
IloOplModelDefinition class
deprecated constructor **51**
IloOplModelSource class
deprecated constructors **51**
IloOplRunConfiguration class
deprecated method **51**
IloOplSettings class
deprecated methods **51**
incompatibilities
between 5.1 and earlier **72**
input/output files, path resolution **72**
integers
not for conditions **57**
inter, OPL keyword **57**
invoke, OPL keyword **67**

J

Japanese IDE **67**
Java
CPLEX callbacks **43**

K

key, OPL keyword **78**
keys in tuple declarations
transitivity of properties **57**
keywords
deprecated after 5.1 **64**
invoke **67**
key **78**
new in 5.1 **67**
prepare **67**
with **78**

L

labeled assertions **40**
language
changes from 5.0 to 5.1 **67**
changes from 5.1 to 5.2 **57**
changes from 5.2 to 5.5 **40**
changes from 5.5 to 6.0 **24**
Launch oplrun, button **39, 42**
localization **67**
logical constraints
in 5.1 **79**
supported by the CP engine **49**
logical expressions **79**

M

memory management and allocation
in flow control script **41**
menu bar
changes in 5.0 **84**
changes in 5.1 **69**
Model Browser
changes in 5.0 **85**

N

nonlinear constraints
in 5.2 **49**

O

ODM
as supported by OPL **75**
operators
and/or, aggregation **57**
inter/union, aggregation **57**
OPL IDE **49**
oplrun **49**
executing projects **81**
in 5.1 **70**
launching from the IDE **30, 42**
on AIX platforms **88**
or operator **57**
ordered sets **40**
Outline window **85**
Output window
changes in 5.0 **86**
changes in 5.1 **68**

P

path resolution in 5.1 **72**
performance
in 5.1 **66**
in 5.2 **54**
performance tuning tool **38**
ports
in 5.1 **66**
in 5.2 **54**
postprocessing
feasible solutions **43**
precedes constraint
not supported in 5.2 **49**
prepare, OPL keyword **67**
Problem Browser
in 5.1 **68**
processing order
for assert and execute statements **57**
Profiler output tab
content and usability in 5.1 **68**
projects
and ILOG Script **80**
creating
in the IDE **87**
executing **87**
from oplrun **81**
Projects window
changes in 5.0 **85**
changes in 5.1 **68**

R

range expressions
ground boundaries **79**
relaxations
control by means of ILOG Script **80**
requires constraint
not supported in 5.2 **49**
run configurations
and ILOG Script **80**
packaging for Customer Support **69**

S

Scheduler
not supported in 5.2 **49**
scripting
changes in 5.5 **41**
in data files **67**
set expressions
aggregation of inter/union **57**
setCoef methods **41, 43**
setLinearCoef method **43**
settings files **87**
sharing data between models **41**
solution pools **38**
sorted sets **40**
specialized constraints **49**
Start window
in 5.0 **85**

T

table loading **67**
terminology **56**
toolbars
changes in 5.0 **85**
changes in 5.1 **68**
tooltips **68**
Tune model, button **39**
tuning process **38**
tuple keys **78**
tuple maps
in Problem Browser **68**
tuple sets
in Problem Browser **68**

U

union OPL keyword **57**

V

Vista
and OPL IDE **58**

W

warm start
fine control **80**
Windows Server 2003 **17**
with, OPL keyword **78**